

CLAIMS

We claim:

1. A method comprising:

receiving a list of candidate primitive actions comprising primitive actions related to a change in a file system state of a first file system, and primitive actions related to a change in a file system state of a second file system, both file systems constituting replicas of a shared file system; and

generating a schedule of non-conflicting primitive actions comprising one or more primitive actions from the list.

2. A method as recited in claim 1 wherein the generating a schedule comprises:

selecting a candidate primitive action from the list based on a value assessment of the candidate primitive actions.

3. A method as recited in claim 1 wherein the generating a schedule comprises:

selecting a candidate primitive action from the list based on a value assessment of the candidate primitive actions; and

executing the selected candidate primitive action starting from a checkpoint file system state; and

1 scheduling the selected candidate primitive action if the executing the
2 selected candidate primitive is successful.
3

4 4. A method as recited in claim 1 further comprising:
5 receiving a log constraint representing a relationship between two of the
6 primitive actions of the first file system, or between two of the primitive actions of
7 the second file system.
8

9
10 5. A method as recited in claim 4 wherein the log constraint comprises
11 a parcel relationship, a predecessor-successor relationship, or an alternatives
12 relationship.
13

14
15 6. A method as recited in claim 1 further comprising:
16 receiving an object constraint representing a relationship between
17 one of the primitive actions of the first file system and one of the primitive actions
18 of the second file system.
19

20
21 7. A method as recited in claim 6 wherein the object constraint
22 comprises a mutually-exclusive relationship or a best-order relationship.
23
24
25

1 8. A method as recited in claim 6 wherein the object constraint is based
2 on application semantics related to the primitive actions of the shared file system.

3
4 9. A method as recited in claim 1 further comprising:
5 proposing the generated schedule to a user.

6
7
8 10. A method as recited in claim 1 further comprising:
9 committing the generated schedule on at least one of the first file
10 system and the second file system.

11
12 11. A method as recited in claim 1 further comprising:
13 selecting one of the scheduled primitive actions to undo; and
14 undoing the selected scheduled primitive action.

15
16
17 12. A method as recited in claim 1 further comprising:
18 selecting one of the scheduled primitive actions to undo;
19 undoing the selected scheduled primitive action; and
20 undoing another of the primitive actions that depends on the selected
21 scheduled primitive action to undo.
22
23
24
25

1 13. A method as recited in claim 1 wherein the receiving the list
2 comprises:

3 decomposing a file system command at the first file system into the
4 primitive actions related to a change in a file system state.
5

6 14. A method as recited in claim 1 wherein the generating comprises:
7 selecting a candidate primitive action from the list;
8
9 executing the selected candidate primitive action starting from a
10 checkpoint file system state;

11 determining whether the executing the selected candidate primitive
12 action was successful; and
13

14 if the executing the selected candidate primitive action was
15 successful, adding the selected candidate primitive action to the schedule.
16

17 15. A method as recited in claim 14 wherein the generating further
18 comprises:

19 if the executing the selected candidate primitive action was
20 successful,
21

22 removing the selected candidate primitive action from the list,

23 removing from the list another candidate primitive action that
24 may execute only before the selected candidate primitive action, and
25

removing from the list another candidate primitive action that
conflicts with the selected candidate primitive action.

16. A method as recited in claim 14 wherein the generating further
comprises:

if the executing the selected candidate primitive action was not
successful,

removing the selected candidate action from the list, and
removing from the list all actions that are in a parcel with the
selected candidate action.

17. A method as recited in claim 14 wherein the generating further
comprises:

if the executing the selected candidate primitive action was not
successful,

rolling back side effects that resulted from executing the
selected candidate action, and

rolling back side effects that resulted from previously
executing other actions that are in a parcel with the selected candidate action.

18. A method comprising:
receiving a first file system command;
receiving a second file system command;
decomposing the first file system command into one or more
corresponding first primitive actions;
decomposing the second file system command into one or more
corresponding second primitive actions;
receiving an object constraint indicating a relationship between a
selected one of the first primitive actions and a selected one of the second
primitive actions; and
if the object constraint indicates mutual exclusion, scheduling either
the selected one of the first primitive actions or the selected one of the second
primitive actions in a schedule of non-conflicting primitive actions based on the
object constraint
otherwise, scheduling both the selected one of the first primitive
actions and the selected one of the second primitive actions.

19. A method as recited in claim 18 further comprising:
executing the selected one of the first primitive actions and the
selected one of the second primitive actions on a file system with the object

1 constraint to determine whether the selected first primitive action and the selected
2 second primitive action conflict.
3

4 20. A method as recited in claim 18 further comprising:
5 logging the first primitive actions in a first action log;
6 logging the second primitive actions in a second action log;
7 combining the first action log and the second action log into a
8 reconciliation log;
9 selecting a primitive action from the reconciliation log; and
10 executing the selected primitive action on a file system.
11
12

13 21. A method as recited in claim 20 wherein the selecting a primitive
14 action comprises:
15 selecting a primitive action from the reconciliation log that has a
16 higher value than the other primitive actions in the reconciliation log.
17
18

19 22. A method as recited in claim 18 wherein the relationship indicated
20 by the object constraint comprises a mutually exclusive relationship.
21
22

23 23. A method as recited in claim 18 further comprising:
24
25

1 identifying a primitive action that conflicts with the scheduled
2 primitive action among the first primitive actions and the second primitive actions;
3 and
4 excluding the identified conflicting primitive action from the
5 schedule of non-conflicting primitive actions.
6
7

8 24. A method as recited in claim 21 wherein the selecting a primitive
9 action comprises:

10 attributing a higher value to a first primitive action than a second
11 primitive action if scheduling the first primitive action would result in fewer
12 conflicts with other primitive actions in the reconciliation log than would
13 scheduling the second primitive action.
14
15
16
17
18
19
20
21
22
23
24
25

1 25. A processor-readable medium having processor executable
2 instructions for performing a method comprising:
3 receiving a first file system command to change the state of a first file
4 system;
5 generating a plurality of first primitive actions corresponding to the first file
6 system command; and
7 receiving one or more log constraints representing a relationship between
8 two of the plurality of first primitive actions; and
9 scheduling one or more of the first primitive actions in a non-conflicting
10 schedule of primitive actions based on the one or more log constraints.
11

12
13 26. A processor-readable medium as recited in claim 25 wherein the
14 method further comprises:
15 receiving a plurality of second primitive actions corresponding to a second
16 file system command to change the state of a second file system;
17 selecting one of the first primitive actions and one of the second primitive
18 actions;
19 receiving an object constraint representing a relationship between the
20 selected first primitive action and the selected second primitive action; and
21 scheduling the selected first primitive action or the selected second
22 primitive action based on the object constraint.
23
24
25

1 27. A processor-readable medium as recited in claim 25 wherein the one
2 or more log constraints comprise user constraints.

3
4 28. A processor-readable medium as recited in claim 25 wherein the one
5 or more log constraints comprise application constraints.

6
7
8 29. A processor-readable medium as recited in claim 25 wherein the
9 method further comprises committing the schedule of non-conflicting actions on
10 the first file system.

11
12 30. A processor-readable medium as recited in claim 26 wherein the
13 object constraint depends upon application semantics.
14
15
16
17
18
19
20
21
22
23
24
25

31. A system comprising:

an input/output module receiving a file system command causing a tentative update to a file system state;

a reconcilable file system operable to receive the file system command, and generate a plurality of primitive actions representing the file system command and a log constraint representing a relationship between two primitive actions in the plurality of primitive actions; and

a log receiving the plurality of primitive actions and the log constraint.

32. A system as recited in claim 31 wherein the reconcilable file system comprises:

a decomposition module operable to decompose the file system command into the plurality of primitive actions; and

a recording module operable to receive the plurality of primitive actions and record the plurality of primitive actions in the log.

33. A system as recited in claim 32 wherein the decomposition module is further operable to generate the log constraint and communicate the log constraint to the recording module.

1 34. A system as recited in claim 32 wherein the log constraint is a user
2 constraint.

3
4 35. A system as recited in claim 32 wherein the log constraint is an
5 application constraint.

6
7
8 36. A system as recited in claim 32 wherein the log constraint is a parcel
9 constraint indicating that the two primitive actions must be executed together.

10
11 37. A system as recited in claim 32 wherein the log constraint is a
12 predecessor-successor constraint indicating that the two actions must be executed
13 in a prescribed order.

14
15
16 38. A system as recited in claim 32 wherein the log constraint is an
17 alternatives constraint indicating that only one of the two actions must be selected
18 and executed.

19
20
21 39. A system as recited in claim 31 wherein the reconcilable file system
22 is further operable to receive a schedule of non-conflicting primitive actions and
23 commit the non-conflicting primitive actions to the file system.

1 40. A system as recited in claim 39 wherein the reconcilable file system
2 is further operable to roll back changes that resulted in the tentative file system
3 state in order to commit the schedule of non-conflicting primitive.
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25